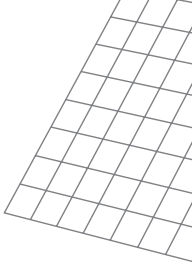


blaxxun Avatar Studio

Build Your Own Virtual Identity in Cyberspace!



User's Guide



© 2000 blaxxun interactive, Inc. All rights reserved.

blaxxun Avatar Studio—User's Guide

blaxxun, the blaxxun logo, blaxxun Contact, Instant Community are registered trademarks of blaxxun interactive.

Avatar Studio is a trademark of CANAL+.

All other trademarks are the property of their respective owners.

First edition. May, 2000.

North America	Europe
blaxxun interactive, Inc. 1550 Bryant Street / Suite 770 San Francisco, CA 94103 USA	blaxxun interactive AG Elsenheimerstr. 61-63 D-80687 München Germany

Table of Contents

blaxxun Avatar Studio	5
IMPORTANT NOTICE!	5
System Requirements	6
Installation	7
Starting the Installation	7
License Key	7
DirectX Setup	8
After the Installation	8
Options in the Start Menu	8
 Getting Started with blaxxun Avatar Studio	 9
Starting the Program	9
Menus	10
Keyboard and Mouse Controls	12
Making a Basic Avatar	14
 Customizing Your Avatar	 15
Adding and Removing Clothing	15
Note	16
Changing Your Avatar's Appearance	16
Changing the Avatar's Physique	17
Changing Your Avatar's Skin	17
Giving Your Avatar a Face	18
Changing Facial Features	18
Using a Photo	19
Animating Your Avatar	19
Step-by-Step Animation Tutorial	21
Step 1	21
Step 2 a	22
Step 2 b	23
Step 2 c	23
Step 2 d	24
Step 2 e	24
Saving Your Avatar	25
Exporting to a VRML File and Uploading to a Community	27

VRML Export	27
Upload to a Community	28

Create an Avatar: Step-by-Step Tutorial	29
--	-----------

Appendix A: Adding Enhanced Animations	34
---	-----------

Implementing a Walk Animation	34
Creating Animations in blaxxun Avatar Studio	34
Preparing the Avatar File	35
Editing the Avatar file	35
Implementing Idle Gestures	39
Creating Idle Gestures in blaxxun Avatar Studio	39
Editing the Avatar File	39
Implementing a Walk Animation - Summary	43
Implementing an Idle Animation - Summary	45

Appendix B: Troubleshooting, Credits and Legal Notices ...	48
---	-----------

Troubleshooting	48
Credits and Acknowledgments	48
Trademarks, Copyright and Warranty	49

blaxxun Avatar Studio

Build Your Own Virtual Identity in Cyberspace!

Welcome to blaxxun Avatar Studio!

With blaxxun Avatar Studio, you can create your own personal avatar, a 3D character that you can use to represent yourself in 3D virtual communities. You decide every detail of your avatar's appearance: body, hairstyle, clothes, etc. To make it more realistic, you can also animate it with lifelike gestures. You can even give it your own face!

Avatars created with blaxxun Avatar Studio are based on the 3D VRML language standard, which allows you to use your avatars in any virtual world that uses this language, e.g. blaxxun's Cybertown or CANAL+'s Le Zeme Monde.

IMPORTANT NOTICE!

Read this before you or your child start using Avatar Studio.

Some people may fall into unconsciousness or suffer from epileptic seizures when exposed to certain types of flashing lights or other common features of everyday life. These people may display symptoms of epilepsy when watching TV programs with flashing images or playing certain types of video games. This may also include people who have never suffered from epileptic fits or have any previous medical history of this kind. However, if you or someone in your family has already displayed symptoms related to epilepsy (fits or falling into unconsciousness) when exposed to light stimuli, please consult a doctor before using this product. We strongly recommend that parents keep an eye on their children when they play video games.

If you or your child have any of the following symptoms, stop playing immediately and consult a doctor.

- vertigo or dizziness
- eye trouble
- contractions or spasms (eyes or muscles)

- falling into unconsciousness
- orientation trouble
- convulsions

Take the following precautions when playing video games:

- Don't stay too close to the screen.
- Play on a computer with a small screen.
- Avoid playing when tired or if you have been losing sleep.
- Play in a well-lit room.
- Take frequent breaks. Five to ten minutes per hour is recommended.

System Requirements

Since blaxxun Avatar Studio is a demanding graphics application, your computer will need to meet certain performance requirements if the software is to work effectively with it. The lists below show the minimum system configuration required for working with blaxxun Avatar Studio, and a recommended system configuration for optimum performance:

Minimum Configuration

- IBM PC or compatible computer
- Pentium MMX processor, 200 MHz
- 32 MB of RAM
- 2X CD-ROM drive
- 2D/3D graphics accelerator, optionally with 3dfx Voodoo add-on card
- Windows 95 or 98 and DirectX 6 or higher or Windows 2000 (comes with DirectX 7)

Recommended Configuration

- IBM PC or compatible computer
- Pentium II processor, 233 MHz
- 64 MB of RAM
- 2X CD-ROM drive
- 2D/3D graphics accelerator with 4 MB graphics memory, optionally with 3dfx Voodoo2 add-on card

- Windows 95 or 98 and DirectX 6 or higher or Windows 2000 (comes with DirectX 7)

Installation

Installing blaxxun Avatar Studio is fast and easy. Most of what you need to know is in the setup program's on-screen instructions. However, please read the following information about situations you could encounter during the installation, and the Start menu options that are available after the installation.

Also, please check the *README* file on the blaxxun Avatar Studio CD for any information about last-minute changes that may have occurred after this documentation was written.

Starting the Installation

To start the installation, just put the blaxxun Avatar Studio CD in your CD-ROM drive and the setup program will start automatically and display a menu. After selecting blaxxun Avatar Studio from the menu, follow the on-screen instructions.

If no menu appears:

- Click on the *Start* button at the bottom of the screen.
- Select *Run ...*
- Enter *D:/setup* (or substitute the appropriate letter for your CD-ROM drive for *D*) and click on *OK*; this will start the installation program.

License Key

If you didn't buy blaxxun Avatar Studio in a blaxxun store, you will see a dialog prompting you to enter a license key. Click on the *Buy* button to buy a license key on the blaxxun website, or click [here](#).

You can run blaxxun Avatar Studio without a license key, but you won't be able to create and upload your VRML avatar for use in a blaxxun community.

But you can always buy a license later and activate it:

- Click on the *Start* button at the bottom of the screen.
- Select *Programs* → *blaxxun Avatar Studio* → *License*.

DirectX Setup

The setup program (CD-ROM installation only) offers you the option to install DirectX 6.1 if it is not already installed on your computer. If you skip the DirectX installation, you can install it later:

- Click on the *Start* button at the bottom of the screen.
- Select *Run ...*
- Enter *D:/directx/dxsetup* (or substitute your CD-ROM's drive letter for *D*) and press *Enter*.

The download version of blaxxun Avatar Studio doesn't include a DirectX setup. You can get one from Microsoft.

After the Installation

After the installation is complete, the setup program gives you an opportunity to start Avatar Studio. If you do, the program's start screen with the message "Loading in progress ..." will appear. Loading may take a few seconds.

Options in the Start Menu

The setup program creates a new *blaxxun Avatar Studio* entry in the Start menu under Programs. It also creates an icon on the Windows desktop that you can use to start the program.

When you click on the *blaxxun Avatar Studio* option in the *Start → Programs* menu, you'll see a menu with the options described below.

blaxxun Avatar Studio	Starts blaxxun Avatar Studio.
Documentation	Displays the HTML-based documentation in your browser.
License	Opens a dialog from which you can buy a license if you don't already have one, or enter the code for a license you've already bought.
My Avatars	Opens the Windows Explorer and displays the files in your Avatars folder.
My Photos	Opens the Windows Explorer and displays the files in your Photos folder.
Select a video card	If you have more than one video card installed, you can use this option to select the card that you want to use with blaxxun Avatar Studio.
UnInstall blaxxun Avatar Studio	Select this option to remove blaxxun Avatar Studio and all of its files from your computer.

GETTING STARTED WITH BLAXXUN AVATAR STUDIO

Starting the Program

After you have installed blaxxun Avatar Studio, you can start the program by either of the methods below:

- Click on *Start*, then select *Programs* → *blaxxun Avatar Studio* → *blaxxun Avatar Studio*.
- Double-click on the *blaxxun Avatar Studio 1.0* icon on the Windows desktop.

It may take a few seconds for the program to load. While the program is loading, the opening screen will be displayed with the message "Loading in progress ..."

After the program finishes loading, the Avatar Studio appears with the default avatar and a dialog in which you can change some basic features to create your own simple avatar. The dialog and other standard features of the Avatar Studio user interface are described below.



1. The (–) control allows you to minimize the Avatar Studio application within Windows. The cross allows you to quit the program. Though pictured here, these controls only become visible after you have specified the avatar's features in (2) below.
2. The dialog box allows you to create an avatar quickly. The default settings are displayed in yellow. You can choose between a male or female; African, Asian or Caucasian appearance; business or casual dress; and the avatar's height and build. Click on *NEXT* to confirm your choice.
 - Note: The features described below only work after you have specified a basic avatar (in item 2 above).
3. The avatar stands on a small pedestal. To view the avatar in profile or from behind, click on the arrows and drag the cursor to the left or right

while holding down the left mouse button. (You can also use the left and right arrow keys on your keyboard.)

Click on *File* to access the following options:

Back	closes the <i>File</i> menu; you can also use the <i>Esc</i> key.
New	to create a new avatar.
Open	to load a previously saved avatar.
Upload	to export a finished avatar to a <i>VRML</i> file and upload it to a virtual community.
Save	to save an avatar.
Stop/Start sound	toggles sound effects on/off.
Exit	to close the application.

4. The menu bar is where you can select the options for customizing your avatar. To select an option, simply click on it. There are 7 options:

Body	Select skin color, pattern, and tattoo. Also specify characteristics of arms, legs, etc.
Face	Select earrings and adjust hairstyle and facial features.
Underwear	Shorts, panties, bras, etc.
Clothing	Various tops and bottoms that you can mix and match.
Outfits	Items such as jackets that are worn on top of items from the Clothing wardrobe.
Accessories	Shoes, hats, jewelry ...
Animation:	Define lifelike (or not so lifelike) movements and gestures.

5. The top segment of the menu pedestal shows which menu is currently active. Use the arrows to move from one menu to another. The bottom segment of the pedestal (Wardrobe) takes you “downstairs” (see below).

Menus

After you have confirmed the features of your basic avatar, the only immediate change in the user interface is the appearance of the options for the *Body* menu. On the menu pedestal (1) you can now see the options *Skin*, *Measurements*, *Chest*, *Arms*, *Behind* and *Legs*. Click on one of the options to open a dialog for modifying the avatar’s settings.



By clicking on *Wardrobe*, you can go “downstairs” where you can select clothes and other items for your avatar. The items available in the wardrobe depend on which main menu option is active, i.e. displayed on the menu pedestal.



1. By clicking on the arrows you can switch to any other main menu option (Body, Face etc.).
2. You can also select an option from the main menu bar by clicking on it.
3. Click on *Custom* to go back “upstairs” where you can modify the avatar and change details of its body, clothes, etc.
4. By clicking on the arrows, you can browse through all the samples available in the currently selected wardrobe.
5. Here you can toggle between male and female wardrobes and select items (clothes, accessories, shoes etc.) from either wardrobe. For example, if you have selected a female avatar, you can still dress her with clothes and accessories available in the male wardrobe, and vice versa.

Keyboard and Mouse Controls

You can use the keyboard and the mouse to perform a variety of actions within Avatar Studio, such as viewing your avatar from different angles and moving through the menu options.

The following keyboard functions are available:

Keys from 0 to 9 (at the upper part of the keyboard)	Launch an animation.
Space bar	Returns the avatar to an upright, standing position.
F11	Make a screen shot; it will be stored in TGA file format in the directory in which you installed Avatar Studio.
F5	Randomize the settings in a dialog.
Up/Down arrow keys	To view the avatar from different angles in a vertical plane.
Left/Right arrow keys	To view the avatar from different angles in a horizontal plane.
PageUp/PageDown	To access the <i>Custom</i> and <i>Wardrobe</i> modes.
Home	To go back to the first option on the menu bar.
End	Takes you to the last option on the menu bar.
Return	Confirms an active dialog.
Esc	Cancels an active dialog or quits the program.
Keys 2, 4, 6, 8 (numeric keypad)	Modify the position of the central animation cursor on two axes.
Keys 7, 9 (numeric keypad)	Modify the position of the central animation cursor on the third axis.

Tip

You can press a vertical and a horizontal arrow key at the same time. In this way, you can view the avatar from different angles while making it rotate on the podium.

You can use your mouse to navigate through the menus and dialogs, change settings, and select or remove objects. In the illustrations below, the numbers refer to entries in the lists below the illustrations. Unless otherwise noted, all actions involve using the left mouse button.



1. To close a window, click on the yellow cross in the upper right corner of the window.
2. Click on + or – to increase or decrease the value of a setting. You can also do this by clicking on the yellow triangle and dragging it to the right or left.
3. Click on an option to make its settings appear, here *Measurements*.
4. Click on an item (for example the trousers) to remove it. The same technique is used to remove any item added to the default avatar, such as an imported image, hair, underwear, clothes and accessories.
5. By clicking on the arrows, you can go from one option (*Body*, *Face*, ...) to another.
6. To select a menu, click on it in the menu bar.
7. Click here to return to the *Wardrobe* mode.
8. Click on *File* to pop up the *File* menu.



1. To move the control panel, click on its title (here *Measurements*) with the left mouse button and hold it down while moving the mouse. To increase or decrease the size of the control panel, click on the title with the right

mouse button and hold it down while moving the mouse up or down. These techniques can be used on all control panels in the Avatar Studio.

2. To increase or decrease the value of a feature or item, move the yellow cursors by holding down the left mouse button and dragging them to the left or right.
3. To remove an item, click on it while holding down your left hand mouse button and drag it toward the middle of the screen.
4. To make the avatar rotate, click on the arrows and move the cursor to the left or right.

Making a Basic Avatar

You can create your own simple avatar with a few mouse clicks. Indeed, if you're happy with the default avatar (a European female of average height and build with casual dress), all it takes is a click on *NEXT* (6 in the illustration below), or you can press the *Enter* key.



You can also choose among the following features before confirming your choice by clicking on *NEXT*.

1. Gender: male or female,
2. Appearance: African, Asian or Caucasian
3. Height: small, medium or tall
4. Build: slim, medium or large
5. Dress: business or casual

Now you have a basic avatar. Move along to the next chapter to learn how to customize it to your own tastes.

CUSTOMIZING YOUR AVATAR

This chapter explains the most important features of blaxxun Avatar Studio, those that let you give your avatar its own “personality”. You'll learn how to dress your avatar, define its physical appearance, give it a face (your own or somebody else's), and create animations that allow your avatar to perform gestures. And once you've created and customized an avatar, you can save it and export it to a *VRML* file for use in blaxxun's virtual communities. That's explained here too.

Adding and Removing Clothing

After you've created a basic avatar, your screen will look like the illustration below: your avatar will have default clothing, and the Body menu will be displayed.

You may want to remove the default clothing before adding items of your own choice. To remove an item, simply click on it (1) or click and drag it toward the middle of the screen.



To add an item or accessory, activate an appropriate menu (e.g. Outfits) by using the arrow keys (2) or clicking on an option (3). Then switch to Wardrobe mode by clicking on *Wardrobe* (4) or pressing PageDn. In the wardrobe you'll see a selection of outfits from which you can choose any combination of tops and bottoms.

When you find a top that you like, click on it (1) to see it on your avatar. Click on the lower part of an outfit (2) to add a skirt or trousers to your avatar.



At any time while dressing your avatar, you can:

- scroll through the wardrobe using the arrows (3),
- view the Male or Female wardrobe by clicking on the corresponding option (4), or
- switch to another main menu option (e.g. *Accessories*) with the arrows (5).

You may think the selection of colors in the wardrobes isn't very good. But after you have chosen an item of clothing for your avatar, you can use the *PageUp* key to move back to the *Outfits* menu, where the new item now appears. Click on the item in the *Outfits* menu to open a control panel with which you can change the color and pattern of the item.

The description above involved the *Outfits* menu. Other menus for dressing your avatar are *Underwear*, *Clothing* and *Accessories*. The latter option is where you can choose items including shoes, hats, jewelry, handbags and glasses. All of these menus work like the *Outfits* menu.

Note

Your avatar will wear items selected from the *Outfits* menu on top of items from the *Clothing* menu, so don't be surprised to see *Clothing* items partly or completely covered by items you select from the *Outfits* menu.

Changing Your Avatar's Appearance

blaxxun Avatar Studio gives you a wide range of options for specifying the physical appearance of your avatar. Features that you can configure include various bodily attributes as well as facial features.

After you have created a basic avatar, the *Body* menu will be active. You can use the options *Measurements*, *Chest*, *Arms*, *Behind* and *Legs* to change the physique of your avatar. In addition, you can specify skin color and pattern with the *Skin* option.

Changing the Avatar's Physique

To modify the avatar's general build, click on *Measurements* (1). A control panel (2) will appear, with slider controls for the different settings. To modify a feature, click on the corresponding yellow triangle and move it to the left or the right by dragging it with the mouse, or click on the + and – controls at the ends of the slider.



To see the effects of your changes, you can view the avatar from different angles by clicking on the arrows below the avatar (3) or by using the left and right arrow keys on your keyboard. Click on the cross in the upper right corner of the control panel (6) to close it.

Though the description above was for the *Measurements* menu, the same applies for the *Chest*, *Arms*, *Behind* and *Legs* options. Experiment with these options until your avatar looks the way you want it to.

Changing Your Avatar's Skin

You have a couple of choices for specifying the appearance of your avatar's skin. The simplest is to switch to *Wardrobe* mode while the *Body* menu is active. In the wardrobe you'll find models with several skin colors; click on one of them to make your selection. You can also choose a tattoo from the wardrobe.

A wider variety of skin colors and patterns is available from the *Skin* option of the *Body* menu. Click on *Skin* to make a control panel appear.



You can choose a skin texture pattern by clicking on one of the small squares in the control panel (2). This illustration shows the *Classic* palette; you can view other palettes by clicking on the arrows to the right and left of *Classic* (3).

If you change your mind about your selection and want to go back to the default skin type, just click on the cross in the upper left corner of the control panel (7).

Adjust the brightness and transparency of your selection with the slider controls (4 and 6). If you'd prefer a skin color that's not available from the wardrobe, experiment by moving the box (5) around on the Color diagram.

Click on the cross (8) in the upper right corner to close the Skin control panel.

Giving Your Avatar a Face

Changing Facial Features

You can change your avatar's facial features in the same way you change its physique. Activate the *Face* menu and use its options to display control panels for changing the avatar's ears, eyes, mouth, cheeks and nose. Use the *Proportions* menu to change size and shape of the entire head and face, and the *Hair-style* option to change hair color and volume.

Switch to *Wardrobe* mode to select hairstyle and earrings for women or hairstyle and beard/mustache for men. If you select an item from the wardrobe, it will be added to the *Face* menu so you can click on it there and customize it.

Using a Photo

A special option of the *Face* menu lets you replace the face provided by blaxxun Avatar Studio with a photo of your choice. The photo should be stored as a *.JPG file in the the *Photos* subdirectory of the directory in which you installed blaxxun Avatar Studio.



In the *face* menu, click on *Picture* (1), then on *Import* (2) to select a photo. To adjust the photo to the face of the avatar, experiment with the active zones (3) that represent the different parts of the face: the outline of the face, eyebrows, eyes, nose and mouth.

Click on one of the red outlines (it will then turn blue) and drag it to a new position, where it will be surrounded by a green box. Click and drag the corners of the green box to change the shape and size of the object.

To view the result, click on *Update* (2) to put the new face on the avatar. If you're not happy with the result (and you'll probably need a few tries before you are), experiment with the active zones and the *Mirror* option (4). With *Mirror:none*, all active zones are used to map the image to the avatar's face. *Mirror:right* uses the values from the right side of the avatar's face and assigns their mirror image to the left side; *Mirror:left* does the opposite.

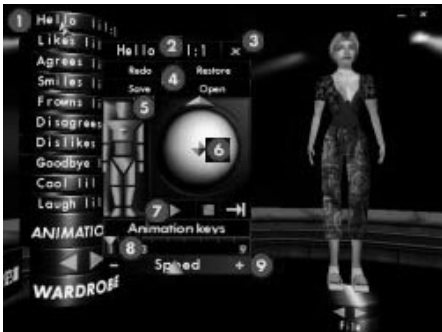
To return the active zones to their original positions (where they were after the picture was imported), click on *Reset*. Click on *Export* to save the result of an Update to a TGA file.

Animating Your Avatar

To define animations for your avatar, activate the *Animation* menu by clicking on it in the *main* menu or pressing the *End* key. The *Animation* menu includes ten animations numbered 0–9 that you can customize for your avatar.



You can keep the default animations or create your own. To find out how to make your own, click on *Hello* (2). The control panel for creating animations, the keyframer, opens. You can still select a different animation by clicking on it in the *Animation* menu.



The top of the keyframer (2) displays information about the active animation. In this example, the animation is *Hello* and has the number “0”. You can launch the animation at any time by pressing the “0” on your keyboard (not on the numeric keypad). To close the control panel, click on the cross (3).

Tip

You can use the arrow keys to view the avatar from different angles.

Four options (4) are available for accessing the files in which the animations are stored:

- Redo clears the current animation. All animation keys (8) (if any are defined) will disappear.
- Save will save the animation in progress.
- Open allows you to load a previously saved animation.

Restore cancels any changes you made since opening the current animation.

The body diagram (5) shows the parts of the avatar that you can animate. Just select the body part that you want to animate; a gray icon will appear there. You can make it move along two axes by using the central animation cursor (6); use the arrow just above the central animation cursor for the third axis.

The playback controls (7) are similar to those on a CD player's control panel. Click on the triangle to play an animation. Click on the square to stop an playback. The icon at the right toggles continuous playback on and off.

The animation key list (8) shows the keys that compose a movement. An animation consists of several different movements, or keys. For example, an animation can move the head and the arm at the same time. Each animation can have up to 9 keys. Each key can include movements for any of the 16 parts of the avatar's body.

To create a key, click twice directly on the part of the body (5) that you want to animate.

Tip

To copy and paste a key from one position to another, click and drag the key to the place you want.

You can control the speed of an animation with the yellow Speed cursor (9). It acts on all movements simultaneously.

This was just a general description of a keyframer. Below is a step-by-step introduction to creating an animation.

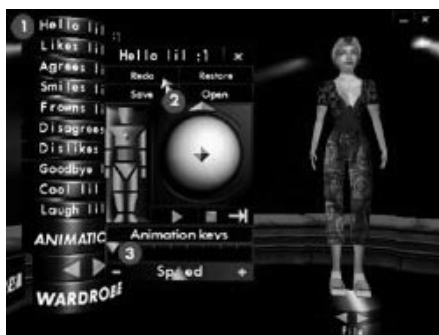
Step-by-Step Animation Tutorial

Step 1

Click on an *animation* (1) then on *Redo* (2) to clear the existing animation settings.

blaxxun Avatar Studio

Customizing Your Avatar – Step-by-Step Animation Tutorial



As you can see in (3) above, there are no animation keys.

Step 2 a

Since we want to keep the starting position of the illustration above, we position the animation key cursor to the second key (1).

Then choose the part of the body that you want to animate. For this example, click on the upper arm (2). Then move the central animation cursor towards the top as shown (3). Then you'll see:



Here you can see that the keys 1 and 9 are created automatically. Why? The keyframer makes an extrapolation between two positions to calculate a movement. This procedure requires two keys.

Copy key number 2 to the positions 4, 6, and 8.

Copy key number 1 to the positions 3, 5, and 7.

Step 2 b

Do the same with the other arm:

- Click on the 2nd key (1),
- Select the part of the body you want to animate (2),
- Move the arm by moving the central animation cursor (3).



Copy key number 2 to the positions 4, 6, and 8.

Copy key number 1 to the positions 3, 5, and 7.

Step 2 c

Now move on to the legs. Perform the following steps:

- Click on the 2nd key (1),
- Select the part of the body you want to animate, in this case the upper leg. (2),
- Move the leg by moving the central animation cursor (3).



blaxxun Avatar Studio

Customizing Your Avatar – Step-by-Step Animation Tutorial

Copy key number 2 to the positions 4, 6, and 8.

Copy key number 1 to the positions 3, 5, and 7.

Step 2 d

Execute the following steps:

- Click on the 2nd key (1),
- Select the other leg (2),
- Move the leg by moving the central animation cursor (3).



Copy key number 2 to the positions 4, 6, and 8.

Copy key number 1 to the positions 3, 5, and 7.

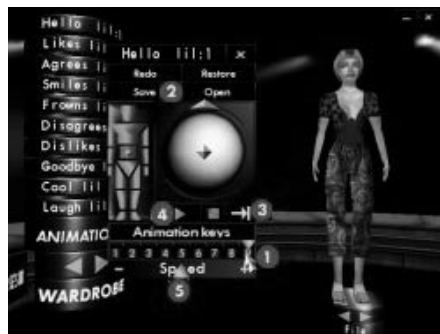
Step 2 e

Click on key number 1 and double-click on the model's stomach (1). Keys number 1 and 9 will appear. Copy key number 1 to positions 3, 5 and 7.

Select the avatar's stomach (1) and click on key number 2 (2) and move it in any direction with the central animation cursor (3).



Copy key number 2 to the positions 4, 6, and 8. Then click on key 9. At this point your screen should look something like this:



You should be able to see all the 9 keys that are part of this animation (1).

To view the result, click on the arrow (3) for continuous playback and then on the play button (4) to play the animation. You can modify the speed of the animation with the cursor (5), and view the animation from different angles by using the four arrow keys on your keyboard (while the animation is playing).

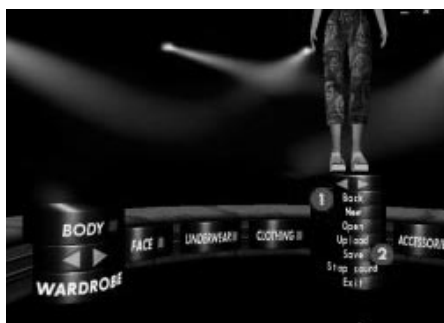
Save your animation by clicking on (2).

Saving Your Avatar

Click on *File* to access the menu below the pedestal (1), then click on *Save* (2).

blaxxun Avatar Studio

Customizing Your Avatar – Saving Your Avatar



A dialog box will appear:



Type a name of your choice on the line with the small blinking yellow cursor (1) and confirm by clicking on *OK* (2) or by pressing *Enter*.

You can close this dialog at any time by clicking on the cross in the upper right corner (3) or by pressing the *Escape* key.

If the name you entered already exists, another dialog box will appear:



If you select **YES** (1), the previously saved avatar will be erased and replaced by the new one. If you don't want this to happen select **NO**. The previous dialog box will then appear, giving you the opportunity to change the name of your new avatar.

You can close this dialog box at any time by clicking on the cross in the upper right corner (2) or by pressing the *Escape* key.

Exporting to a VRML File and Uploading to a Community

VRML Export

Before you can use your new avatar in a virtual world, you'll need to create a *VRML* file using blaxxun Avatar Studio, and then upload the *VRML* file to one of the blaxxun communities that Avatar Studio is configured to connect with.

Go to the *File* menu and click on *Upload*. A dialog box (1) will appear asking you to enter the name of the file. Confirm the export by clicking on *OK*. It may take a few seconds to complete the *VRML* export process.



The export process creates two files with the name you specified above; they are stored in the Avatars subdirectory of the directory in which you installed blaxxun Avatar Studio. The *VRML* file has the *.wrl* suffix and there is also an image (*.jpg*) file. Note that if you want to distribute your avatar, both of these files must stay together. So if you want to use your avatar in a community that Avatar Studio doesn't connect to, or just want to share it with friends, remember that you need to transfer both files.

Note

You can't change the *VRML* files using Avatar Studio. If you want to change an avatar, use the *File/Open* option and select one of your saved avatars. Specialists wanting to look at the *VRML* source can do so by adding the suffix "gz" to the avatar's filename and then unzipping it with the gzip utility.

Upload to a Community

After Avatar Studio creates and saves your avatar's *VRML* files, it begins the process of uploading your avatar to one of blaxxun's communities. First it opens a browser window and displays a web page prompting you to select a community for your avatar. Choose a community and click on the Upload button.

Some communities will request that you provide your blaxxun Avatar Studio license key, so it's a good idea to have that ready when you begin to upload to a community for the first time. After the first upload, the community stores your license information and won't ask for it again during future uploads.

Note

If you want to avoid starting the upload process every time you save an avatar to *VRML*, you can rename the file *upload.exe* in the Avatar Studio directory.

CREATE AN AVATAR: STEP-BY-STEP TUTORIAL

This chapter is a quick and easy tutorial to guide you through some of the steps involved in creating your own avatar. It doesn't show you everything, but should give you a good idea of what can be done.

To begin, start blaxxun Avatar Studio and create a default avatar. After you've done that, your screen should look like the illustration below:



Since the first steps will involve changing the avatar's physique, begin by removing the clothes (and the hair) from the default avatar until only the underwear is left. That will make it easier to see the effects of changes you'll make in the next steps. How to remove the clothes? Simply click on an item to remove it. Or click on an item and drag it toward the middle of the screen.

Click on *Measurements* (1) to modify the avatar's body (height, build etc.). The Measurements control panel is illustrated below.



You can modify the size or form of a feature by clicking on + or – or by dragging the yellow cursor (2) to the left or right.

Tip

To get a better view of the effects of your modifications, use the arrows to zoom in on the avatar or to view it from different angles. If you want to make random settings for all values in a control panel, press the *F5* key.

Now click on the *Chest* option (1). Modify the shape or the size by moving the yellow cursor to the left or right (2). Some modifications are better viewed in profile; use the arrows to turn the avatar on its pedestal (3).



You can modify the legs and the behind in the same way. Now click on *Wardrobe* (4) to go on to the next step.

Use the arrows (1) to activate the *Face* menu, where you can select a hairstyle by clicking on one of the hairstyles displayed in the wardrobe (2):



To modify the hairstyle click on *Custom* (3), which will take you to the *Face* menu where you can customize the hair and facial features.

In the *Face* menu, click on *Hairstyle* (1) and customize the hairstyle using the available controls.



Here you can set the volume and the hair color (2), including any natural or unnatural color that strikes your fancy.

The same technique can be used for the different parts of the face, including eyes, mouth, cheeks, ears and nose. The *Proportions* option lets you modify the size and shape of the entire face.

When you've completed this step, return to *Wardrobe* mode (3) where you'll select clothes for your avatar.

Go to the *Underwear* section of the wardrobe by using the arrows (1):



Select a bodysuit by clicking on it (2) and then click on *Custom* to change its appearance. Now the *Underwear* menu is active. Click on *Body* (1) to specify the fabric and transparency of the body:



Choose a color by sliding the square around in the color box (2) and choose a pattern (3). Then return to the wardrobe (4) for the next step. Use the arrows (1) to move to the *Outfits* section.



Select a top (2) and skirt (3). Then return to *Custom* mode (4).

Click on *Skirt* (1) in the *Outfit* menu to open a control panel:



Modify the skirt by changing the color (2) and the fabric (3), then do the same for the top. After the clothes suit your taste, switch back to *Wardrobe* mode (4).

Use the arrows (1) to move to the *Accessories* section:



Use the arrows (2) to find shoes (3) and then a handbag and other accessories. Then click on *Custom* (4) to give them a more personal touch. The *Accessories* menu is now active. Click on Handbag (1) to change the color and fabric of your handbag; you can do the same for your shoes.



Modify the accessories by changing the color (2) and the fabric (3).

As you can see, there's a wide range of possibilities for building and dressing your avatar; these examples only scratch the surface of what's possible. We invite you to experiment and design your own individual avatar that's as unique as you are.

APPENDIX A: ADDING ENHANCED ANIMATIONS

Though avatars created with blaxxun Avatar Studio have a relatively realistic appearance and are capable of performing lifelike gestures, you'll notice that they don't perform a walk animation when they move. And, unless you constantly activate gestures via chat macros, they just stand around like lifeless dummies.

In this appendix, we describe how to add a walk animation to add realism to your avatar when it moves from place to place in a virtual world. And, to add some life to it when it's not moving or performing gestures on your orders, we show you how to add idle gestures that the avatar will perform when it's not doing anything else.

This appendix begins with detailed descriptions of the walk and idle animations. These are followed by brief summaries that you can use as a quick reference if you've already implemented the animations once or twice before.

Note: The enhancements described below are relatively sophisticated and involve making changes to the VRML code that defines the avatar. You should only attempt them if you are familiar with VRML and the concepts of implementing interactive behavior by adding scripts to VRML code.

Implementing a Walk Animation

After designing an avatar with blaxxun Avatar Studio and uploading it to a virtual world, you'll notice that your avatar doesn't perform a walk animation when moving; it just glides from one position to the next. Walk animations are not supported in the current version of blaxxun Avatar Studio. This brief tutorial will show you how to add a walk animation to your avatar. It assumes that you are familiar with VRML and the concepts of implementing interactive behavior by adding scripts to VRML code.

Creating Animations in blaxxun Avatar Studio

When defining gestures for your avatar, we recommend defining a walk animation as the ninth and a stop animation as the tenth gesture in Avatar Studio. This is because gestures 1 to 8 will be triggered by chat macros in blaxxun Contact. blaxxun Avatar Studio allows you to define ten gestures, leaving two more gestures that aren't triggered by blaxxun Contact chat macros. The default animation names for those two gestures are "cool" and "laugh." You can trigger these two gestures with chat keywords in Contact. If you want to keep these gestures rather than replacing them with the walk and stop gestures, edit the avatar as described below in the section "Implementing Idle Gestures." In the interest of

keeping the file size as small as possible, this tutorial will show you how to define "walk" as ninth and "stop" as tenth and make sure that these new gestures aren't triggered by keywords from the chat.

Designing a walk animation is a difficult and demanding task. Keep in mind that your avatar's movements must be loopable in order to get a smooth animation.

You will need to define a gesture as stop animation as well. This animation is necessary to bring all extremities back to their original positions so your avatar won't stop and remain suspended in some phase of a walking movement. So basically your stop animation doesn't have to be an animation in the sense of moving some extremity. But you still have to use Avatar Studio to set keys in the first and ninth frames to the default position for all animatable extremities. This is the only way to make sure that your avatar returns to its standard posture when it stops walking. In other words, playing back this gesture will show your avatar standing still, not making any movements. But this is exactly what you want to have happen when your avatar stops walking.

Preparing the Avatar File

First of all, you'll need to unzip the avatar using gzip. Gzip is a common compression tool, which is distributed under GNU license (<http://www.gzip.org/>). You can download a DOS version of gzip from <http://www.blaxxun.com/developer/gzip.exe>. In order to reduce the download avatar file size, blaxxun Avatar Studio output uses gzip compression. Unzip your avatar using the following command:

```
gzip -d avatarfile.wrl
```

After unzipping the avatar file, you can edit it with any ordinary text editor.

Editing the Avatar file

In the PROTO interface, you will find an eventIn SFVec3f set_position statement. When you are moving through a 3D scene, your avatar will send its changed position to the community server, which will distribute this event to all avatars in the scene. Since your avatar is also a member of the scene, it will receive this event through the set_position event in the PROTO interface and will then move to the new position. That's why we will use this event to trigger our walk animation.

```
PROTO Avatar [  
#more events are listed here  
eventIn SFVec3f set_position  
#more events are listed here  
]
```

The exported avatar file is already prepared for the addition of interactive behavior like walk animations. At the top of the file you can see a script node called `AvatarScript`. This script makes sure that the avatar geometry follows your current position in virtual space. But it also triggers several time sensors that you can find below. In our case, we only care about the `TimeStop` time sensor, which is triggered when no movement is necessary anymore. So we'll use it to trigger our stop animation.

After the data for the geometry, you'll find a bunch of `TimeSensor` and `Interpolator` statements. Those nodes describe the animations for the gestures you defined. Find the definition of the time sensor that is responsible for the walk animation. That should be

```
DEF TS9 TimeSensor{cycleInterval 1.3333}
```

If you defined `gesture9` as the walk animation, edit this line so that it looks as follows:

```
DEF TS9 TimeSensor{cycleInterval 1.3333 loop TRUE enabled FALSE}
```

This is because we need a looped animation for continuous movement, but we don't want it to be active from the beginning on; that's why the time sensor has to be disabled. We will add a script for enabling and disabling this timer now; the script is shown below. Copy this source code and paste it into your avatar file, preferably before the `ROUTE` statements at the end of the file.

```
DEF wa_script Script {
  eventIn SFVec3f set_position IS set_position
  eventIn SFBool set_watched
  eventIn SFBool stop_walk
  field SFNode walk USE TS9
  field SFNode stop USE TS10
  field SFBool watched FALSE
  field SFVec3f dest 0 0 0
  exposedField SFBool isStop TRUE
  field SInt32 stopp 0
  field SFNode gest USE Welder
  directOutput TRUE
  url "vrmlscript:
  function set_watched(v,t){watched = v;}
  function set_position(v, t)
  {
    if(dest[0] != v[0] || dest[1] != v[1] || dest[2] != v[2])
    {
      isStop = false;
      stopp = 0;
      if(stop.isActive){stop.stopTime = t;}
```

```

if(!walk.enabled && watched)
{
walk.enabled = true;
gest.G8 = t;
}
dest = v;
}
}
function stop_walk(b, t)
{
if(!b)
{
if(walk.enabled)
{
walk.enabled = false;
walk.stopTime = t;
}
    if(stopp < 1)
    {
stop.startTime = t;
isStop = true;
stopp++;
}
return;
}
}"}

```

The first eventIn statement (SFVec3f set_position IS set_position) refers to the set_position (described earlier) that is received whenever the avatar has to be moved to a new position. In the corresponding VRMLScript method, the current position is compared with the latest received position. A difference between the values means that the avatar has to be moved and therefore has to perform a walk animation. In that case, the time sensor for the walk animation gets enabled and started. Note that the time sensors for walk and stop animations are referred directly, which means that we manipulate the nodes directly without routing.

```

field SFNode walk USE TS9
field SFNode stop USE TS10

```

If your walk and stop animations are not triggered by the time sensors TS9 and TS10, you have to refer here to the ones that do trigger them. In that case you would have to edit the following line as well:

```
gest.G8 = t;
```

This line starts the desired time sensor; gest refers to another script in the avatar file which is responsible for correct mesh movements. G8 is a function in that

script that starts a time sensor. If your walk animation is defined as gesture 3, then you would have to edit this line as follows:

```
gest.G2 = t;
```

As you see, our script node contains a second VRMLScript method that is responsible for stopping the walk animation and starting the stop animation. Because of a little delay in event processing, we have to do this in a separate method which is not triggered by the `set_position` event. As described earlier, your client software sends your current position to the server. This happens 3 times per second when you're moving, but it only happens once every 10 seconds when you're not moving. So your avatar would continue performing its walk animation as long as it hasn't received the same position as it already has, which wouldn't happen until 10 seconds after it stopped moving. To avoid this, we use the `TimeStop` time sensor from the `AvatarScript`. This timer gets activated when you stop moving. So routing the `isActive` event of this time sensor to the script will trigger the second VRMLScript function. This function basically stops the walk timer and disables it and starts the stop timer in order to reset all avatar extremities. In order to do this, add the following `ROUTE` statement to the end of your avatar file:

```
ROUTE TimeStop.isActive TO wa_script.stop_walk
```

The avatar only needs to perform an animation when it's visible; for this reason, the avatar file already contains a visibility sensor. To use this in the script, the script node contains an `eventIn SFBool set_watched` statement. The corresponding VRMLScript function sets the `watched` flag to `TRUE` if the avatar is visible; only in this case does it perform the walk animation. Therefore, you also need to add a route from the visibility sensor to the script:

```
ROUTE vis.isActive TO wa_script.set_watched
```

Finally, since it would be undesirable to have the walk and stop animations be triggered by anything other than movement, there is one last thing to do. In the avatar file, find the node called `Welder`. This is a script node that is responsible for the correct movement of every vertex in the avatar mesh. In the script interface of this script node, there are ten `eventIn SFTIME` statements named `G0` to `G9`. These events are responsible for triggering the start of a specified gesture. As you see, these events are mapped by the `IS` statement to the `PROTO` interface. This means that they directly receive external events. To prevent the walk and stop gestures from being started by external calls like `blaxxun Contact chat` macros, remove the `IS` mappings in the `eventIns G8` and `G9`, which are responsible for starting the two animations.

Your avatar will now perform a walk animation whenever you move in virtual space. To reduce the download size, compress your avatar file again. Use the following `gzip` command:

```
gzip -9 avatarfile.wrl
```

Since gzip will automatically rename the file to *avatarfile.wrz*, be sure to rename it to the original name afterward.

Implementing Idle Gestures

At this point, we have a humanoid figure that is able to make several movements (gestures 1 to 8) and perform a walk animation. But as soon as it stops walking in the virtual world, the avatar becomes a stiff, lifeless dummy if you don't activate any gestures. The avatar would look more realistic if it could automatically perform gestures during idle time. The following tutorial will show you how to add such functionality.

Creating Idle Gestures in blaxxun Avatar Studio

Idle gestures are additional gestures to those you defined while creating your avatar with Avatar Studio. This means that, in order to keep the existing gestures, we need to define new gestures, export the avatar using a different name, and copy the appropriate data from the resulting "dummy" avatar to the original avatar file.

The first thing to do is to load the existing avatar in blaxxun Avatar Studio and create new gestures for the idle animations. blaxxun Avatar Studio allows the user to design up to ten gestures. After creating the new gestures, upload the avatar. Make sure to use a new filename and not to overwrite the existing avatar. Only parts of this new avatar file will be used and inserted in the existing avatar file.

Note: Keep in mind that additional animation data will increase the file size of your avatar, and think about how much data you want to make other community members download everytime you enter a virtual world.

Editing the Avatar File

Unzip your avatar file (as described in the previous section) and open it in your text editor. We will now insert the animation data for the newly created idle gestures into the existing avatar file. These data are time sensors, position interpolators and orientation interpolators. Every Avatar Studio avatar is divided into 42 rotation centers. Each of these controls one part of an extremity, e.g. upper arm, lower arm, hand etc. Depending on the gestures you defined, the VRML file contains key positions and orientations for these animated centers. In the existing avatar file, there are already ten gestures defined; these have nodes with the standard names. That's why we need to rename the time sensors and interpola-

tors for the idle gestures in the dummy avatar file; otherwise we would have duplicate node names in the existing avatar file, which would end up causing strange side effects. So rename all nodes named TSx to TSx+10, e.g. TS1 to TS11, TS2 to TS12 and so on. Do the same for PI1 to PI10 and OI1 to OI10.

After renaming all the necessary nodes in the dummy avatar file, copy the corresponding data and paste it into the existing avatar file just before the script node called **Welder**. (**where?**) For example, let's assume you have designed 3 gestures for idle times. You now have to copy all data from the code block starting with:

```
DEF TS11 TimeSensor{}
```

and ending with the last ROUTE statement of TS13, e.g.

```
ROUTE TS13.fraction_changed TO OI13_-1.set_fraction
```

Paste this data in the existing avatar file, just above the **Welder** script.

Now let's look at the **Welder** script node in the existing avatar file. This is the most important node in the avatar file, because it is responsible for correctly positioning every single point in the avatar mesh. So be careful when making changes to this node.

In the script interface of this node, you'll find ten `eventIn SFTIME` statements, named G0 to G9. These events trigger the start of a gesture. Because we're now adding gestures to the avatar file, we need to add events to this script interface as well. So add as many `eventIn SFTIME` statements as you have idle gestures (3 in our example) and name them G10, G11 and so on.

Farther below in the script interface of this node, you'll find a `field MFNode ts` statement. This is an array of nodes; its elements are the gesture time sensors. Refer to the time sensors for your idle gestures (the ones you just added) in this array as well. The array should now look like this:

```
field MFNode ts[USE TS1,USE TS2,USE TS3,USE TS4,USE TS5,USE
TS6,USE TS7,USE TS8,USE TS9,USE TS10,USE TS11,USE TS12,USE
TS13]
#add as many time sensors as you have for idle gestures
```

This script node contains several VRMLScript functions that make complex point transformations. We don't need to go into details about that now. But look for the following section in the script node:

```
function G0(t){if(watched)Gstart(0,t);}
...
function G9(t){if(watched)Gstart(9,t);}
```


These are the functions that correspond to the `SFTime` events that were mentioned above and are called when the corresponding event occurs. If the avatar is visible, then the corresponding gesture is started by calling another function (`Gstart`) with the index of the time sensor and the current time stamp as parameters. To have our idle gestures start this way, we need to add similar VRML-Script functions here:

```
function G10(t){if(watched)Gstart(10,t);}  
function G11(t){if(watched)Gstart(11,t);}  
function G12(t){if(watched)Gstart(12,t);}
```

The effect of these functions is that if the event `G10` (which we added to the script interface above) occurs, the function `Gstart` gets called, which instantly starts the eleventh time sensor in the `ts` field.

This was all we had to edit in `Welder`. Now we need to make sure that this script receives all the necessary events. To do this, you need to find the appropriate `ROUTE` statements at the end of the dummy avatar file. For instance, something like:

```
ROUTE PI11_1.value_changed TO Welder.P1  
ROUTE OI11_1.value_changed TO Welder.rot_Center  
...  
ROUTE TS13.isActive TO Welder.Gend
```

Copy all the necessary routes (all routes from position interpolators and orientation interpolators of idle gestures to `Welder`) and paste them in at the end of the existing avatar file.

We have now prepared our avatar file and can start to program the idle functionality. We don't want the avatar to perform these gestures unless nothing else is happening, especially not when the avatar is walking. Also, to improve performance, we only want to start these gestures if the avatar is visible. For these reasons, we need to coordinate the activities with the walk animation. That's why we will insert our functions into the script node that controls the walk animation, which we inserted earlier.

First of all we need a timer that triggers the script node. Insert the following time sensor node after the `Welder` script node:

```
DEF random_clock TimeSensor{cycleInterval 8 }
```

Now we need to make changes to our `wa_script`, which is responsible for the walk animation so far. Add the following lines to the script interface:

```
field SFTime lastTime 0  
eventIn SFBool set_active
```

```
eventOut SFTime randomTime_changed
```

To perform avatar animations during idle time, we'll use a random function. This will keep the avatar from continuously repeating the same series of gestures; instead it will perform them in random order, which should look more realistic. The random choice of an idle gesture will be done in the following VRMLScript function; add it to the `wa_script` script node.

```
function set_active(v,t){
  if(v){return;}
  if(!watched || !isStop){randomTime_changed = t; return;}
  rnd = Math.random() * 100;
  if(rnd <25){randomTime_changed = t; return;}
  if(rnd >=25 && rnd < 35){gest.G10 = t;}
  if(rnd >=35 && rnd < 45){gest.G11 = t;}
  if(rnd >=45 && rnd < 55){gest.G12 = t;}
  if(rnd >=55 && rnd < 65){gest.G10 = t;}
  if(rnd >=65 && rnd < 80){gest.G11 = t;}
  if(rnd >=90){gest.G12 = t;}
  randomTime_changed = t;
}
```

The idea behind this function is simple. It always gets called when our random timer sends an `isActive` event, which happens when the time sensor starts and stops. If `isActive` is `true` (meaning that the time sensor is running) the function will not perform anything. But when the time sensor stops, the event sends `false`. In that case, the function first checks whether the avatar is visible and whether the walk animation is stopped. If so, it computes a random number. Depending on this number, a gesture is started and the random clock is started again. This way we can make sure that an idle gesture is only performed every 8 seconds, because this is the cycle interval the random clock is running. If you want to change this interval, change the `cycleInterval` field in the `random_clock` node. The numerical value of this field is declared in seconds.

The gestures are started by a direct call to the appropriate function in `Welder`, e.g. `gest.G10 = t` directly sets the current time as input for the eventIn `G10` in `Welder`. There, this time is passed as `startTime` to the eleventh time sensor in the node array `ts` in `Welder` (we made changes to this array earlier). This means that in case you have more idle gestures and have inserted more time sensors in the `MNode` array in `Welder`, you will have to change these calls in this function.

To ensure that the random clock runs as soon as the avatar file is loaded, we need to define an `initialize` function to start the clock. It looks like this:

```
function initialize(){
  randomTime_changed = Browser.getTime();
}
```

Insert this function into the `wa_script` script node as well.

Finally, we have to add the routes from the random clock to the script and back so that both communicate with each other.

```
ROUTE wa_script.randomTime_changed TO  
random_clock.set_startTime  
ROUTE random_clock.isActive TO wa_script.set_active
```

After that the avatar is more realistic and can not only perform walk animations, but apparently self-triggered idle movements as well. To reduce the download size, compress your avatar file again with this `gzip` command:

```
gzip -9 avatarfile.wrl
```

Since `gzip` will automatically rename the file to *avatarfile.wrz*, be sure to rename it to the original name afterward.

Implementing a Walk Animation - Summary

1. Create your avatar with Avatar Studio.
2. Define a walk animation as ninth gesture and a stop animation as tenth gesture.
3. Upload the avatar.
4. Unzip the avatar using `gzip`:

```
gzip -d avatarfile.wrl
```

5. Edit the time sensor for the walk animation so that it loops and is disabled:

```
DEF TS9 TimeSensor{cycleInterval 1.3333 loop TRUE enabled  
FALSE}
```

6. Copy the following script and paste it into the avatar file before the `ROUTE` statements at the end of the file.

```
DEF wa_script Script {  
eventIn SFVec3f set_position IS set_position  
eventIn SFBool set_watched  
eventIn SFBool stop_walk  
field SFNode walk USE TS9  
field SFNode stop USE TS10  
field SFBool watched FALSE  
field SFVec3f dest 0 0 0
```

```

exposedField SFBool isStop TRUE
field SFInt32 stopp 0
field SFNode gest USE Welder
directOutput TRUE
url "vrmlscript:
function set_watched(v,t){watched = v;}
function set_position(v, t)
{
if(dest[0] != v[0] || dest[1] != v[1] || dest[2] != v[2])
{
isStop = false;
stopp = 0;
if(stop.isActive){stop.stopTime = t;}
if(!walk.enabled && watched)
{
walk.enabled = true;
gest.G8 = t;
}
dest = v;
}
}
function stop_walk(b, t)
{
if(!b)
{
if(walk.enabled)
{
walk.enabled = false;
walk.stopTime = t;
}
if(stopp < 1)
{
stop.startTime = t;
isStop = true;
stopp++;
}
return;
}
}"}

```

7. Make sure that the walk and stop fields from the script interface refer to the time sensors that trigger the walk and stop animations via USE statement, i.e.

```

field SFNode walk USE TS9
field SFNode stop USE TS10

```

8. Add the following ROUTE statements below the script node that you just pasted.

```
ROUTE TimeStop.isActive TO wa_script.stop_walk  
ROUTE vis.isActive TO wa_script.set_watched
```

9. In the Welder script node, find the eventIn statements G8 and G9 and remove the IS statements after them.
10. Compress the file using the following gzip command:


```
gzip -9 avatrfile.wrl
```
11. Since gzip will automatically rename the file to *avatarfile.wrz*, be sure to rename it to the original name afterward.

Implementing an Idle Animation - Summary

1. Load the avatar in Avatar Studio.
2. Create animations for idle gestures.
3. Upload the avatar, using a different filename from that of the avatar you want to keep.
4. Unzip the "dummy" avatar using gzip:


```
(gzip -d avatarfile.wrl)
```
5. Rename all time sensors in the dummy file from TSx to TSx+10, i.e. TS1 to TS11, TS2 to TS12 and so on.
6. Rename all position and orientation interpolators according to this principle, i.e. PI1 to PI11 and OI1 to OI11 and so on
7. Copy all data from the first time sensor to the last ROUTE statement of your idle animations, i.e. from DEF TS11 TimeSensor{} to ROUTE TS13.fraction_changed TO OI13_-1.set_fraction.
8. Paste this data into the "real" avatar file below its animation data.
9. Now continue working in the real avatar file.
10. In the Welder script node, find the eventIn SFTIME statements G0 to G9 and add as many of those eventIn statements as there are idle gestures and name them correspondingly, e.g. G10 and so on.
11. Find a field MFNode ts statement in the Welder script node and add the appropriate time sensors to trigger the idle gestures, i.e.

```
field MFNode ts[USE TS1,USE TS2,USE TS3,USE TS4,USE TS5,USE
TS6,USE TS7,USE TS8,USE TS9,USE TS10,USE TS11,USE TS12,USE
TS13]
```

12. Find the following section in the Welder script node:

```
function G0(t){if(watched)Gstart(0,t);}
...
function G9(t){if(watched)Gstart(9,t);}
```

13. Add as many similar functions here as you have idle gestures, and name them according to the eventIn statements you just added (see step 10), e.g.

```
function G10(t){if(watched)Gstart(10,t);}
function G11(t){if(watched)Gstart(11,t);}
function G12(t){if(watched)Gstart(12,t);}
```

14. From the dummy avatar file, copy all the ROUTE statements that route from position and orientation interpolators to the Welder script and insert them below the ROUTE statements in the real avatar file at the end of the PROTO implementation, i.e.

```
ROUTE PI11_1.value_changed TO Welder.P1
ROUTE OI11_1.value_changed TO Welder.rot_Center
...
ROUTE TS13.isActive TO Welder.Gend
```

15. Add the following time sensor above the wa_script script node (which you added while implementing the walk animation):

```
DEF random_clock TimeSensor{cycleInterval 8 }
```

16. Add the following lines to the script node interface (the lines before the url statement) in wa_script:

```
field SFTIME lastTime 0
eventIn SFBool set_active
eventOut SFTIME randomTime_changed
```

17. Add the following functions to the wa_script script node:

```
function set_active(v,t){
if(v){return;}
if(!watched || !isStop){randomTime_changed = t; return;}
rnd = Math.random() * 100;
if(rnd < 25){randomTime_changed = t; return;}
if(rnd >= 25 && rnd < 35){gest.G10 = t;}
if(rnd >= 35 && rnd < 45){gest.G11 = t;}
if(rnd >= 45 && rnd < 55){gest.G12 = t;}
if(rnd >= 55 && rnd < 65){gest.G10 = t;}
```

```
if(rnd >=65 && rnd < 80){gest.G11 = t;}  
if(rnd >=90){gest.G12 = t;}  
randomTime_changed = t;  
}  
function initialize(){  
randomTime_changed = Browser.getTime();  
}
```

18. Edit this function according to your needs. `gest.G10` to `gest.G12` send a direct call to a function in `Welder` which starts the appropriate gesture, i.e. `gest.G10` will start the eleventh time sensor in the `ts` field in `Welder`, which should be `TS11`.

19. Add the following ROUTE statements below the `wa_script`:

```
ROUTE wa_script.randomTime_changed TO  
random_clock.set_startTime  
ROUTE random_clock.isActive TO wa_script.set_active
```

20. Compress the file using the following gzip command:

```
gzip -9 avatrfile.wrl
```

21. Since gzip will automatically rename the file to `avatarfile.wrz`, be sure to rename it to the original name afterward

APPENDIX B: TROUBLESHOOTING, CREDITS AND LEGAL NOTICES

Troubleshooting

Display problems

Most display problems encountered during use of blaxxun Avatar Studio result from the use of outdated drivers for graphics cards. If you have display problems with Avatar Studio, please contact the manufacturer of your graphics card for new drivers. In most cases, you should be able to download the latest drivers from the manufacturer's website.

Multiple graphics cards

blaxxun Avatar Studio automatically detects the video cards that are already installed in your computer. The program will select one of them when it starts. If you don't like the choice, you can always choose another one in *Start* → *Programs* → *blaxxun Avatar Studio* → *Select a video card*.

Missing cursor in Windows

Occasionally you may find that your mouse cursor is missing after you minimize Avatar Studio and return to Windows. Use the Alt-Tab key combination to switch back to Avatar Studio, then repeat and switch to a Windows application. That will usually make the cursor reappear.

Credits and Acknowledgments

blaxxun interactive

Kristof Nast-Kolb, Pascal Baudar, Thilo Schwerdfeger, John Lucas

CANAL+

Stéphane Castaing, Alain Le Diberder, Frédéric Le Diberder, Frédéric Lemaire, Olivier Ryard

Comptoir des Planètes

Didier Bouchon, Sébastien Guilbert, Alexandre Hadjadj, Dominique Hayez, Marcello Morra

Beta testers

Holger Grahm – blaxxun interactive

Cyril Jolit, Vincent Morel, Thomas Veauclin – ISI DEVELOPEMENT

Special thanks to

Alban, Angéline, Joëlle, Sophie, Stéphane, Mathieu, Judith, Vanessa, Guillaume, Frédéric, Laurence.

Trademarks, Copyright and Warranty

This product is protected by copyright. All rights reserved. For private, non-commercial use only. No part of this product may be reproduced or transmitted by any means or in any form without prior consent in writing from blaxxun interactive.

Avatar Studio is a trademark of CANAL+. All product names and trademarks mentioned herein are the property of their respective owners.

Every effort has been made to ensure that the information in this manual is accurate. blaxxun interactive is not responsible for printing or clerical errors. blaxxun interactive accepts no liability for consequences resulting from the use of the blaxxun Avatar Studio CD-ROM under normal conditions of utilization.